rajalakshmicolleges.org

# Week-07-01-Practice Session-Coding: Attempt review | REC-CIS

7-8 minutes

#### **Question 1**

Correct

Marked out of 5.00

#### **Question text**

Sunny and Johnny like to pool their money and go to the ice cream parlor. Johnny never buys the same flavor that Sunny does. The only other rule they have is that they spend all of their money.

Given a list of prices for the flavors of ice cream, select the two that will cost all of the money they have.

For example, they have m = 6 to spend and there are flavors costing cost = [1, 2, 3, 4, 5, 6]. The two flavors costing 1 and 1 meet the criteria. Using 1-based indexing, they are at indices 1 and 1.

# **Function Description**

Complete the code in the editor below. It should return an array containing the indices of the prices of the two flavors they buy.

It has the following:

240701008

· m: an the amount of to spend

integer denoting money they have

cost: an integer array denoting the cost of each flavor of ice cream

## **Input Format**

The first line contains an integer, t, denoting the number of trips to the ice cream parlor. The next t sets of lines each describe a visit. Each trip is described as follows:

- 1. The integer *m*, the amount of money they have pooled.
- 2. The integer n, the number of flavors offered at the time.
- 3. **n** space-separated integers denoting the cost of each flavor: cost[cost[1], cost[2], ..., cost[n]].

**Note:** The index within the cost array represents the flavor of the ice cream purchased. **Constraints** 

- $1 \le t \le 50$
- $2 \le m \le 10^4$
- $2 \le n \le 10^4$
- 1 ≤ cost[i] ≤  $10^4$ , " i Î [1, n]
- There will always be a unique solution.

# **Output Format**

For each test case, print two space-separated integers denoting the indices of the two flavors purchased, in ascending order.

## **Sample Input**

2

45

14

53

2

4

4

2243

# **Sample Output**

14

12

## **Explanation**

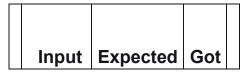
Sunny and Johnny make the following two trips to the parlor:

- 1. The first time, they pool together m = 4 dollars. Of the five flavors available that day, flavors 1 and 4 have a total cost of 1 + 3 = 4.
- 2. The second time, they pool together m = 4 dollars. TOf the four flavors available that day, flavors 1 and 2 have a total cost of 2 + 2 = 4.

Answer:(penalty regime: 0 %)

```
|#include<stdio.h>
    int main()
 2
 3 ₹ {
         int t,m,n,c=0;
 4
         scanf("%d",&t);
 5
         for(int i=0;i<t;i++)</pre>
 6
 7 ₹
 8
             C=0;
             scanf("%d\n%d",&m,&n);
 9
             int arr[n];
10
             for(int j=0;j<n;j++)</pre>
11
12 *
             {
                  scanf("%d",&arr[j]);
13
14
             for(int a=0;a<n-1;a++)
15
16 v
             {
                  for(int b=a+1;b<n;b++)</pre>
17
18 *
                      if(arr[a]+arr[b]==m)
19
20 *
                           printf("%d %d\n",a+1,b+1);
21
22
                           c=1;
                           break;
23
24
25
                  if(c==1)
26
                  break;
27
28
29
         return 0;
30
31
```

## **Feedback**



2	240701008	

2	1 4	1 4	
4	12	12	
5			
1453			
2			
4			
4			
2243			

Passed all tests!

#### **Question 2**

Correct

Marked out of 5.00

### **Question text**

Numeros the Artist had two lists that were permutations of one another. He was very proud. Unfortunately, while transporting them from one exhibition to another, some numbers were lost out of the first list. Can you find the missing numbers?

As an example, the array with some numbers missing, arr = [7, 2, 5, 3, 5, 3]. The original array of numbers brr = [7, 2, 5, 4, 6, 3, 5, 3]. The numbers missing are [4, 6].

#### **Notes**

- If a number occurs multiple times in the lists, you must ensure that the frequency of that number in both lists is the same. If that is not the case, then it is also a missing number.
- You have to print all the missing numbers in ascending order.

240701008

Print each once, even if it is times.

missing number missing multiple

The difference between maximum and minimum number in the second list is less than or equal to **100**.

Complete the code in the editor below. It should return an array of missing numbers.

It has the following:

· arr: the array with missing numbers

brr: the original array of numbers

## **Input Format**

There will be four lines of input:

*n* - the size of the first list, *arr* 

The next line contains *n* space-separated integers *arr[i] m* 

- the size of the second list, brr

The next line contains **m** space-separated integers **brr[i]** 

## **Constraints**

 $1 \le n, m \le 2 \times 10^5$ 

. n ≤ m

 $1 \le brr[i] \le 2 \times 10^4$ 

· Xmax – Xmin < 101

# **Output Format**

Output the missing numbers in ascending order.

240701008

# **Sample Input**

10

203 204 205 206 207 208 203 204 205 206

13

203 204 204 205 206 207 205 208 203 206 205 206 204

# **Sample Output**

204 205 206

## **Explanation**

204 is present in both arrays. Its frequency in arr is 2, while its frequency in brr is 3. Similarly, 205 and 206 occur twice in arr, but three times in brr. The rest of the numbers have the same frequencies in both lists.

Answer:(penalty

240701008

regime: 0 %)

```
#include<stdio.h>
 2
    int main()
 3 ,
    {
 4
         int n,m,c,c1=0,co;
 5
         scanf("%d",&n);
 6
         int arr[n];
 7
         for(int a=0;a<n;a++)</pre>
 8
             scanf("%d",&arr[a]);
 9
10
         scanf("%d",&m);
11
12
         int brr[m],ans[m];
13
         for(int b=0;b < m;b++)
14
         {
             scanf("%d",&brr[b]);
15
16
         for(int j=0;j<m;j++)</pre>
17
18 -
         {
19
             c=0;
             for(int i=0;i<n;i++)</pre>
20
21 .
22
                  if(arr[i]==brr[j])
23
                  {
24
                      c=1;
25
                      arr[i]=-1;
26
                      break;
27
28
29
             if(c==0)
30 -
31
                  ans[c1]=brr[j];
32
                  c1++;
33
34
         for(int a=0;a<c1;a++)
35
36 +
37
             co=0;
38
             for(int b=0;b<c1;b++)
39
40
                  if(ans[b]<ans[a])
41
                 CO++;
42
43
             int temp=ans[a];
44
             ans[a]=ans[co];
45
             ans[co]=temp;
46
47
         for(int i=0;i<c1;i++)
48
         printf("%d ",ans[i]);
49
         return 0;
50
```

9 of 14

### **Feedback**

Input	Expected	Got
10	204 205	204 205
Input	Expected	Got
203 204 205 206 207 208 203 204 205	206	206
206		
13		
203 204 204 205 206 207 205 208		
203		
206 205 206 204		

Passed all tests!

#### **Question 3**

Correct

Marked out of 5.00

#### **Question text**

Watson gives Sherlock an array of integers. His challenge is to find an element of the array such that the sum of all elements to the left is equal to the sum of all elements to the right. For instance, given the array arr = [5, 6, 8, 11], 8 is between two subarrays that sum to 11. If your starting array is [1], that element satisfies the rule as left and right sum to 0.

You will be given arrays of integers and must determine whether there is an element that meets the criterion.

Complete the code in the editor below. It should return a string, either YES if there is an element meeting the criterion or NO otherwise.

## It has the following:

arr: an array of integers

## **Input Format**

The first line contains *T*, the number of test cases.

The next **T** pairs of lines each represent a test case.

- The first line contains *n*, the number of elements in the array *arr*.
- The second line contains n space-separated integers arr[i] where
   0 ≤ i < n.</li>

## **Constraints**

- · 1 ≤ T ≤ 10
- $1 \le n \le 10^5$
- $1 \le arr[i] \le 2 \times 10^4$
- · 0 ≤ i ≤ n

# **Output Format**

For each test case print YES if there exists an element in the array, such that the sum of the elements on its left is equal to the sum of the elements on its right; otherwise print NO.

# Sample Input 0

2

3

123

4

1233

Sam	ple	Ou	ıtp	ut	0
-----	-----	----	-----	----	---

NO

YES

# **Explanation 0**

For the first test case, no such index exists.

For the second test case, arr[0] + arr[1] = arr[3], therefore index 2 satisfies the given conditions.

## Sample Input 1

3

5

11411

4

2000

4

0020

# Sample Output 1

YES

YES

YES

# **Explanation 1**

In the first test case, **arr[2] = 4** is between two subarrays summing to **2**.

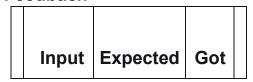
In the second case, arr[0] = 2 is between two subarrays summing to 0.

In the third case, arr[2] = 2 is between two subarrays summing to 0.

Answer:(penalty regime: 0 %)

```
#include<stdio.h>
 2
    int main()
 3 v
 4
         int t,n,Is,rs,m;
         scanf("%d",&t);
 5
         for(int i=0;i<t;i++)
 6
 7 *
 8
             Is=0;
 9
             rs=0;
             scanf("%d",&n);
10
             int arr[n];
11
             for(int j=0;j<n;j++)</pre>
12
             scanf("%d",&arr[j]);
13
             m=n/2;
14
             if(arr[m]==0)
15
16 v
                  for(m=0;arr[m]==0 && m<n;m++);
17
18
             for(int j=0;j<=m;j++)</pre>
19
             Is=Is+arr[j];
20
             for(int j=m;j<n;j++)</pre>
21
             rs=rs+arr[j];
22
             printf("%s\n",(Is==rs)?"YES":"NO");
23
24
25
         return 0;
26
```

## **Feedback**



3	YES	YES	
5	YES	YES	
1141	YES	YES	
4			
2000			
4			
0020			
Input	Expected	Got	
Input 2	<b>Expected</b> NO	<b>Got</b>	
-	•		
2	NO	NO	
2 3	NO	NO	
2 3 123	NO	NO	

Passed all tests!